

AD-A074 932

JOHNS HOPKINS UNIV BALTIMORE MD  
ROBUST WAVEFORM DESIGN. (U)  
SEP 78 H L WEINERT, A H EL-SAWY

DEPT OF ELECTRICAL --ETC F/6 17/2

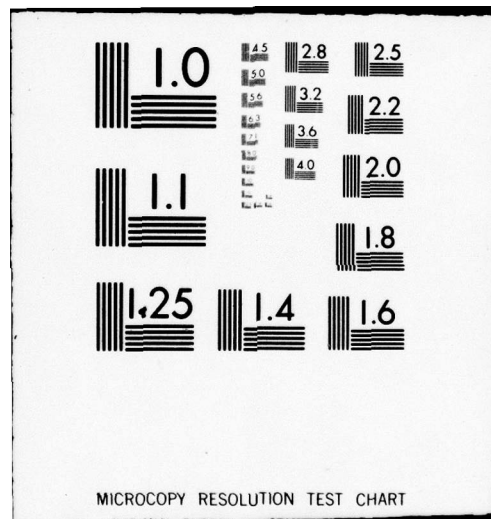
N00014-77-C-0617

NL

UNCLASSIFIED

1 OF 1  
ADA  
074932





JOEL H. MORRIS



# THE JOHNS HOPKINS UNIVERSITY

## LEVEL

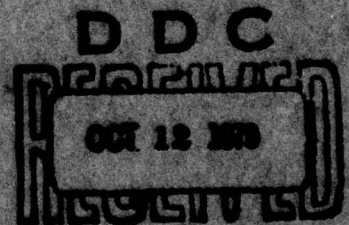
### ELECTRICAL ENGINEERING DEPARTMENT

#### ROBUST WAVEFORM DESIGN

HOWARD L. WEINERT  
ABDEL-RAHMAN H. EL-SAWY

#### FINAL REPORT

CONTRACT N00014-77-C-0617  
NAVAL RESEARCH LABORATORY  
WASHINGTON, D.C. 20375



SEPTEMBER, 1978

APPROVED FOR PUBLIC RELEASE  
RESTRICTIONS UNLIMITED

*Monitored by*  
JOEL H. MORRIS

NAVAL RESEARCH LABORATORY  
WASHINGTON, D.C. 20375

ADA074932

DDC FILE COPY

## **DISCLAIMER NOTICE**

**THIS DOCUMENT IS BEST QUALITY  
PRACTICABLE. THE COPY FURNISHED  
TO DDC CONTAINED A SIGNIFICANT  
NUMBER OF PAGES WHICH DO NOT  
REPRODUCE LEGIBLY.**

-- 1 - AGENCY ACCESSION NO: DN780240  
 -- 2 - DATE OF SUMMARY: 15 FEB 79  
 -- 3 - DATE OF PREV. SUMMARY: 01 AUG 78  
 -- 4 - KIND OF SUMMARY: COMPLETED  
 -- 5 - SUMMARY SECURITY IS: UNCLASSIFIED  
 -- 6 - SECURITY OF WORK: UNCLASSIFIED  
 -- 8A1 - DIST. LIMITATION: UNLIMITED  
 -- 8B - CONTRACTOR ACCESS: YES  
 -- 10A1 - PRIMARY PROGRAM ELEMENT: 61153N  
 -- 10A2 - PRIMARY PROJECT NUMBER: R02105  
 -- 10A2A - PRIMARY PROJECT AGENCY AND PROGRAM: R02105  
 -- 10A3 - PRIMARY TASK AREA: RR0210542  
 -- 10A4 - WORK UNIT NUMBER: R1101102  
 -- 10C1 - CONTRIBUTING PROGRAM ELEMENT (2ND): 37 NRL T  
 -- 10C2 - CONTRIBUTING PROJECT NUMBER (2ND): +CHNOLOGY BA  
 -- 10C3 - CONTRIBUTING TASK AREA (2ND): +E  
 -- 11 - TITLE: (U) HF WAVEFORM SOFTWARE TEST BED  
 -- 11A - TITLE SECURITY: U  
 -- 12 - S + T AREAS:  
 -- 009700 MATHEMATICS AND STATISTICS  
 -- 004200 COMPUTERS  
 -- 021000 RADIO COMMUNICATIONS  
 -- 13 - WORK UNIT START DATE: SEP 77  
 -- 14 - ESTIMATED COMPLETION DATE: OCT 78  
 -- 15A - PRIMARY FUNDING AGENCY: NAVY  
 -- 16 - PERFORMANCE METHOD: CONTRACT  
 -- 17A1 - CONTRACT/GRANT EFFECTIVE DATE: SEP 77  
 -- 17A2 - CONTRACT/GRANT EXPIRATION DATE: SEP 78  
 -- 17B - CONTRACT/GRANT NUMBER: N00014-77-C-0617  
 -- 17C - CONTRACT TYPE: COST TYPE  
 -- 17E - KIND OF AWARD: CON  
 -- 17F - CONTRACT/GRANT CUMULATIVE DOLLAR TOTAL: \$ 15,028  
 -- RESOURCE ESTIMATES  
 -- 18A4 MANYRS CFY-4: + 18B4 FUNDS CFY-4: +  
 -- 18A3 MANYRS CFY-3: + 18B3 FUNDS CFY-3: +  
 -- 18A2 MANYRS CFY-2: + 18B2 FUNDS CFY-2: \$ 15,000  
 -- 18A1 MANYRS CFY-1: + 18B1 FUNDS CFY-1: +  
 -- 18A MANYRS CFY: + 18B FUNDS CFY: +  
 -- 19A - DOD ORGANIZATION: NAVAL RESEARCH LABORATORY (7522)  
 -- 19B - DOD ORG. ADDRESS: WASHINGTON, D.C. 20375  
 -- 19C - RESPONSIBLE INDIVIDUAL: MORRIS, J M  
 -- 19D - RESPONSIBLE INDIVIDUAL PHONE: 202-767-3544  
 -- 19U - DOD ORGANIZATION LOCATION CODE: 1100  
 -- 19S - DOD ORGANIZATION SORT CODE: 33632  
 -- 19T - DOD ORGANIZATION CODE: 251950  
 -- 20A - PERFORMING ORGANIZATION: THE JOHNS HOPKINS UNIVERSITY ELECTRICAL  
 -- ENGINEERING DEPT.  
 -- 20B - PERFORMING ORG. ADDRESS: BALTIMORE, MD 21218  
 -- 20C - PRINCIPAL INVESTIGATOR: WEINERT, H L  
 -- 20D - PRINCIPAL INVESTIGATOR PHONE: 301-338-7032  
 -- 20F - ASSOCIATE INVESTIGATOR (1ST): EL-SAWY, A H  
 -- 20U - PERFORMING ORGANIZATION LOCATION CODE: 2404  
 -- 20N - PERF. ORGANIZATION TYPE CODE: 1  
 -- 20S - PERFORMING ORG. SORT CODE: 25208  
 -- 20T - PERFORMING ORGANIZATION CODE: 400464  
 -- 21E - MILITARY/CIVILIAN APPLICATIONS: CIVILIAN  
 -- 22 - KEYWORDS: (U) LPS VOL-1 (U) COMPUTER SIMULATION (U) ROBUST  
 -- DETECTION (U) INTERFERENCE MODELING  
 -- 23 - TECHNICAL OBJECTIVE: (U) TO DEVELOP A SOFTWARE TEST BED TO EVALUATE  
 -- UNDER GENERAL CHANNEL MODEL ASSUMPTIONS, THE PERFORMANCE OF USER-  
 -- SPECIFIED HF WAVEFORMS, MODULATION, AND RECEIVED SCHEMES

-- 22 - KEYWORDS: (U) LPS VOL-1 (U) COMPUTER SIMULATION (U) ROBUST  
 -- DETECTION (U) INTERFERENCE MODELING  
 -- 23 - TECHNICAL OBJECTIVE: (U) TO DEVELOP A SOFTWARE TEST BED TO EVALUATE,  
 -- UNDER GENERAL CHANNEL MODEL ASSUMPTIONS, THE PERFORMANCE OF USER-  
 -- SPECIFIED HF WAVEFORMS (MODULATION, CODING) AND RECEIVER SCHEMES  
 -- (DETECTION/ ESTIMATION) IN JAMMED AND UNJAMMED ENVIRONMENTS.  
 -- THE TEST BED WILL ALSO BE UTILIZED AS A DESIGN TOOL FOR ROBUST HF WAVEFORMS AND  
 -- RECEIVERS.  
 -- 24 - APPROACH: (U) THE SOFTWARE TEST BED SHOULD BE BASED ON THEORETICAL  
 -- RESULTS VIA MINIMAX OPTIMIZATION AND OTHER ROBUST DETECTION TECHNIQUES  
 -- AND ON SIMULATION (MONTE-CARLO) ALGORITHMS. THE THEORETICAL RESULTS  
 -- SHOULD PROVIDE EQUATIONS TO BE SOLVED BY THE TEST BED FOR SEVERAL  
 -- PERFORMANCE MEASURES (E.G., ERROR PROBABILITY, DETECTION PROBABILITY,  
 -- FALSE-ALARM RATE, ETC.) AND WILL THEN BE COMPLEMENTED BY SIMULATIONS.  
 -- THE TEST BED SHOULD EVENTUALLY ACCOMMODATE FAIRLY-GENERAL DESCRIPTIONS  
 -- OF THE HF WAVEFORMS AND CHANNEL MODELS.  
 -- 25 - PROGRESS: (U) CONTRACTOR HAS COMPLETED THE DESIGN AND DELIVERED A  
 -- SOFTWARE TEST BED WHICH WILL EVALUATE THE PERFORMANCE OF A WIDE CLASS OF  
 -- DETECTORS FOR THE DIFFERENT SIGNAL WAVEFORMS AGAINST ARBITRARY  
 -- INTERFERENCE PROBABILITY DISTRIBUTIONS. THE VARIOUS MODELS OF SIGNALS  
 -- AND INTERFERENCE ARE TREATED AS SUBROUTINES TO BE CALLED BY USER CONTROL.  
 -- THE TEST BED IS CAPABLE OF BEING EASILY EXPANDED OR MODIFIED.  
 -- 37 - DESCRIPTORS: (U) ALGORITHMS (U) CHANNELS (U) SIMULATION  
 -- (U) RATES (U) PROBABILITY (U) MODULATION (U) MODELS (U)  
 -- FREQUENCY (U) FALSE ALARMS (U) ERRORS (U) CODING (U) DOMAINS  
 -- (U) DETECTION (U) COMPUTERIZED SIMULATION (U) COMPUTER PROGRAMS  
 -- 39 - PROCESSING DATE (RANGE): 23 MAY 79  
 --\*\*\*\*\*

-- <<CENTER NEXT COMMAND>>

2

6 ROBUST WAVEFORM DESIGN

10 Howard L./Weinert  
Abdel-Rahman H./El-Sawy  
Principal Investigators  
Department of Electrical Engineering  
The Johns Hopkins University  
Baltimore, Maryland 21218

16 R02105

17 RR0210542

9 Final Report.,  
15 ~~SECRET~~ N00014-77-C-0617  
Naval Research Laboratory  
Washington, D.C. 20375

APPROVED FOR PUBLIC RELEASE  
DISTRIBUTION UNLIMITED

11 Sept ~~1978~~ 1978

1244

Accession For	
NTIS GRA&I	<input checked="checked" type="checkbox"/>
DDC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By _____	
Distribution/	
Availability Codes	
Dist	Avail and/or special
A	231

L000464

## Table of Contents

I.	INTRODUCTION	1
II.	PACKAGE DESCRIPTION	4
II.1	DPPRE.FOR	5
II.2	DPEP.FOR	7
II.3	DPPOST.FOR	8
II.4	The Library DPED	10
II.5	The Library DPESN	11
III.	UPDATING AND EXTENDING THE PACKAGE	13
IV.	PROGRAMS SOURCE LISTING	15
	REFERENCES	

I. Introduction

Detectors and estimators which utilize signals in the high frequency band (HF) as input data usually suffer from performance deterioration as a result of the presence of high frequency interference. This interference can be categorized as either intelligent or non-intelligent. The non-intelligent interference occurs mainly as the result of channel noise, which is primarily impulsive in nature in the HF band (atmospheric and man-made noise) and should be accounted for during the receiver design stage. The intelligent interference is developed by another user either involuntarily when he utilizes the same channel, or deliberately to degrade the receiver performance. The second type of interference in general has a more deleterious affect on the receiver performance than the first and should be taken into account in the process of signal design.

The classical approach for solving this problem is either to design a receiver and to choose the signal which is optimal for a channel model developed apriori from channel measurements, or to utilize an adaptive technique which works for a finite-dimensional class of channel models. A dangerous property of the first approach is that the performance of these optimal receivers usually deteriorates very badly if the actual data did not follow the assumed

channel model exactly [1] and [2]. On the other hand, continuous changes in channel models in HF impulsive noise channels is a known fact for communication engineers, at least for the non-intelligent part of the noise [3],[4], and [5]. Moreover, there is no presently known procedure to protect the receiver completely from the presence of bad data, especially those due to measurement errors either in the process of channel modeling or while taking the observations.

The main difficulty with the adaptive approaches in general is that the communication system designer never knows a priori which parameters in his assumed channel model should be allowed to change with the measurement of the current data.

As a result of the above observations, the development and utilization of new techniques for the design of robust receivers, which depend only on partial information about the channel and provide good performance over all the infinite dimensional class of channels which possess the properties described by this partial information, has received considerable attention during the last few years [6],[7],[8], and [9]. This work deals with the problem of receiver design in the presence of non-intelligent interference. In [6] and [7] the problem of receiver design in the presence of measurement errors has been treated. In [9] a general approach for receiver design when only

partial information about the channel is available has been introduced. The results achieved in this direction are highly encouraging, and invite more effort and research toward the development of similar techniques to overcome the effects of intelligent interference on the system performance. Specifically, it would be of interest to both the communicator and the interferor to know about the worst case interference and the worst case performance of certain receivers and certain signal forms.

This information could be used by the communicator to choose the best signal waveform and detection strategy in the presence of the worst case interference. In other words, to design a system which maximizes, over all combinations of signal waveforms and detection strategies, the worst case performance. The interferor can use this information to prepare an interference strategy which gives maximum degradation of the system performance without apriori knowledge of the signal waveform at each point of time and within his constraints, such as transmitted power limits.

As a compatible effort in this line of research the test-bed package "Detector Performance Evaluation" (DPE) was designed to evaluate the performance of a wide class of detectors for different signal waveforms against arbitrary noise distributions. A detailed description of the contents of this package is given below.

## II. Package Description

The test-bed package DPE in a fortran oriented simulation package which consists of three programs and two supporting libraries. These three programs are DPEP.FOR, DPPRE.FOR, and DPPOST.FOR. The two libraries are called DPED and DPESN. The main program in the package is DPEP.FOR which runs with the support of the above two libraries. The other two programs run independently. No other libraries or subroutines are required other than the standard fortran library. The first program to be executed should be DPPRE.FOR. The main function of this program is to prepare an input file for the main program DPEP.FOR. The name of this file is DPIN.DATA and it contains all required information about the detector to be tested, the signal waveform and the noise distribution. Utilization of the program DPPRE facilitates running the program DPEP off-line. To limit the storage requirements, the output of DPEP will be a binary file which can be translated into a formatted file using the program DPPOST. The library DPED contains 10 detector functions, five of which are already specified and five of which are left to the user to specify. These detector functions are called d1,d2,..., d10. The library DPESN contains both signal waveforms and noise distribution functions. It can accept up to 5 different waveforms called s1,s2,..., s5 and 10 different noise distributions called zn1,zn2,..., zn10. Both these libraries are extendable. A detailed description of these programs and libraries is to be given below.

## II.1 DPPRE.FOR

The main function of this program is to prepare an input data file for the main program DPEP.FOR. The inputs to the program are the answers to a set of questions which appear on the terminal. The output of the program is a file called DPIN.DATA of format (5i5, 5f10.6). This file contains the previous answers in the sequence required by the program DPEP.FOR which utilizes this information to define the detector to be tested, the signal waveform and the noise distribution function and parameters. Beside each question the format of the answer will appear to the user.

The first question to appear is of the form "detector no. = ?". The answer to this question is an integer between 1 and 10 and defines which detector function in the library DPED is to be tested. For example if the answer is 3, then the detector function to be tested is d3 which is the "Limiter-Correlator Detector".

The next two questions define the noise distribution function. The first one will be of the form "noise = ?". The answer to this question defines the noise distribution function. For example, if it is 1 then the noise distribution function will be zn1 which is the normal distribution in the library DPESN. The answer to this question should be an integer between 1 and 10. The answer to the second question which appears in the form "scale = ?" affects the program DPEP in different ways depending on the answer to the previous

question. If the answer to the previous question was 1 or 2 then the answer to this question determines the noise standard deviation. If the previous answer was 3 then it determines the standard deviation of  $\log(x)$ . If the previous answer was 4 or 5 then this number defines the standard deviation of the contaminating distribution; the standard deviation of the contaminated distribution is always one.

The next two questions to appear are signal related questions. The first is about the signal amplitude and has the form "sig. amp. = ?". The second is about the signal waveform and has the form "sig = ?". The answer to this question should be an integer between 1 and 5, and defines which signal waveform from those in the library DPESN should be used. This waveform is also utilized as the reference signal for correlation type detectors.

The next two questions define both the sample size and the number of runs to be averaged. They are of the form "sample size =? " and "no. of runs=1000x?" respectively. The answer to the first one should be a positive integer less than or equal to 30. The second can take any integer value acceptable by the machine.

The next set of questions has the form "Upper Threshold=?" and "Lower Threshold=?" and determines the threshold(s) of the test and the type of test to be performed. If the upper threshold ( $t_u$ ) is equal to the lower threshold ( $t_l$ ), then it is a one threshold test, and the program DPEP determines

the probability that the test statistic is on either side of the threshold as is the case in the minimum probability of error criterion. If  $t_u$  is not equal to  $t_l$ , the distance between  $t_l$  and  $t_u$  will be divided into 500 equal intervals, and the results of DPEP will be in the form of a histogram.

If the answer to question 1 in this program was 4 or 5 then  $t_l$  and  $t_u$  should be between  $\pm 8.0$ .

The last question here is the form "a=?" and it appears only if the answer to question 1 was 5. The answer here defines the break point for the nonlinearity utilized by detector d5. More details for this number will be given later.

## II.2 DPEP.FOR

This is the main program in the test-bed package DPE. It consists of four routines, the main routine and three subroutines d, c and e. It runs with the support of the two libraries DPED and DPESN. The input to this program is the file DPIN.DATA which is the output of the program DPPRE. The output file is an on-record binary file of length (2012) which contains the number of observation groups, the upper and lower thresholds, and the number of runs in which the test statistic falls in some interval.

The main routine and the two subroutines d and c define, respectively, for the subroutine e which detector, noise distribution, and waveform combination is to be tested. The actual performance evaluation test occurs in subroutine e.

This subroutine utilizes the function call name delivered to it by the three other routines and the rest of the information in the input file to calculate the value of the test statistic under the assumption of independent data. If the test is a one threshold test it compares the test statistic with the threshold. If it is a two threshold test, it divides the distance between the two thresholds into 500 equal intervals and decides in which interval the test statistic falls, and adds 1 to the counter for this interval. Every one thousand runs, this subroutine updates the output file.

### II.3 DPPOST.FOR

This program translates the binary file DPEP.DATA, which is the output of the program DPEP, into a formatted file called DPOUT.DATA. The first record in this file contains the number of runs. The second contains two columns; the first contains the threshold and the second contains the probability that the test statistic exceeds this threshold (the probability of false alarm or detection depending on whether the signal is present or not). A sample of this file is shown on the next page.

number of runs =	2000.0	
Threshold (T)		$p(x>T)$
1.75000		0.78799999
1.76000		0.77550000
1.77000		0.76400000
1.78000		0.75349998
1.79000		0.74349999
1.80000		0.73600000
1.81000		0.72600001
1.82000		0.71799999
1.83000		0.70749998
1.84000		0.69900000
1.85000		0.69250000
1.86000		0.68049997
1.87000		0.67150003
1.88000		0.66149998
1.89000		0.65050000
1.90000		0.63700002
1.91000		0.62449998
1.92000		0.61350000
1.93000		0.59700000
1.94000		0.58550000
1.95000		0.57349998
1.96000		0.55849999
1.97000		0.54449999
1.98000		0.53350002
1.99000		0.51349998
2.00000		0.50000000
2.01000		0.48600000
2.02000		0.46950001
2.03000		0.45899999
2.04000		0.44450000
2.05000		0.43000001
2.06000		0.41400000
2.07000		0.40200001
2.08000		0.39050001
2.09000		0.38000000
2.10000		0.37349999
2.11000		0.36300001

## II.4 The Library DPED

This library contains up to 10 detector functions. Five are specified and are briefly described below. Throughout this section we shall consider  $X_i$  as the  $i$ th observation,  $n$  as the total number of observations,  $S_i$  as the  $i$ th signal sample, and  $T_n$  as the test statistic.

### II.4.1 function d1 (Linear detector)

$$T_n = \frac{1}{n} \sum_{i=1}^n X_i$$

### II.4.2 function d2 (correlator detector)

$$T_n = \frac{1}{n} \sum_{i=1}^n S_i X_i$$

### II.4.3 function d3 (Limiter - correlator detector) [10]

$$T_n = \frac{1}{n} \sum_{i=1}^n S_i l(X_i)$$

where

$$l(X_i) = \begin{array}{ll} -1.14 & X_i < -1.14 \\ X_i & |X_i| \leq 1.14 \\ 1.14 & X_i > 1.14 \end{array}$$

### II.4.4 function d4 (M-detector for contaminated normal class) [9]

$T_N$  is such that

$$\sum_{i=1}^N S_i l(X_i - T_N S_i) = 0$$

$l(.)$  is the same as for the previous detector.

#### II.4.5 function d5 (M-detector for p-point class) [9]

$T_N$  is such that

$$\sum_{i=1}^N S_i l(X_i - T_N S_i) = 0$$

If the class is defined by

$$F = \{f : \int_{-\alpha}^{\alpha} f(x) dx \leq 1/2\}$$

then

$$l(t) = \begin{cases} -\tan(c\alpha) & t \leq -\alpha \\ \tan(ct) & |t| \leq \alpha \\ \tan(c\alpha) & t \geq \alpha \end{cases}$$

$$c = \frac{1}{1.718\alpha}$$

Notice that the parameter  $\alpha$  is the same as  $\alpha$  in the program DPPRE.

#### II.5 The Library DPESN

This library consists of 5 signal waveforms and 10 noise distribution functions. Of the five signals only one is specified as a constant signal. Of the ten noise distribution functions the following five are specified. The parameter  $\alpha_n$  in all these functions is the scale factor in the program DPPRE.

II.5.1 function zn1 (normal distribution)

$$f(x) = \frac{1}{\alpha_n \sqrt{2\pi}} \exp \left[ -\frac{1}{2} \frac{(x^2)}{\alpha_n^2} \right]$$

II.5.2 function zn2 (double exponential)

$$f(x) = \frac{1}{2\alpha_n} \exp \left[ -\frac{|x|}{\alpha_n} \right]$$

II.5.3 function zn3 (lognormal component)

$$x = y \cos \theta$$

$$f(y) = \frac{1}{x\alpha_n \sqrt{2\pi}} \exp \left[ -\frac{\log^2(x)}{2\alpha_n^2} \right] \quad y \geq 0$$

$$f(\theta) = \frac{1}{2\pi} \quad |\theta| \leq \pi$$

II.5.4 function zn4 (normal contaminated by normal)

$$f(x) = .9 \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} + .1 \frac{1}{\alpha_n \sqrt{2\pi}} e^{-\frac{x^2}{2\alpha_n^2}}$$

II.5.5 function zn5 (normal contaminated by double exponential)

$$f(x) = .9 \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} + .1 \frac{1}{2\alpha_n} e^{-\frac{|x|}{\alpha_n}}$$

### III. Updating and Extending the Package

1. The maximum sample size allowed in this package is 30. To increase this limit, the user should change the dimension of the arrays  $xd$  and  $ref$  in instruction number 2 in the subroutine of the program DPEP.
2. To add a new detector function -- within the limit of ten detectors -- the user has only to replace one of the detector functions in library DPED by his new function. If we want to replace the detector function  $d_i$  (where  $i$  is an integer between 1 and 10), the call name for the new detector function should be  $d_i(xd, n, y, z)$ , where  $xd$  is an array for the samples,  $n$  is the sample size,  $y$  is an array for the reference signal samples, and  $z$  is an optional variable.
3. Similar methods would be used to update the signal waveforms and the noise functions in the library DPESN.
4. To extend the number of detector functions beyond the limit of 10, the user must change both library DPED and the program DPEP. For example if the user wants to add detector number 11, then a detector function  $d_{11}(x, n, y, z)$  has to be added to the library DPED. Then a statement of the form, `if(i .eq. 11) call d11(idata, dat, d11)` has to be added just before the "END" statement in the main routine of the program DPEP. The external statement in this routine has to be changed to contain `d11`.

5. To extend the number of signal waveforms beyond the limit of 5, the user must add his new waveform to the library DESN. Assuming it is waveform number 6 then the call name for the new function would be S6(j), where j is the order of the signal sample. The next step is to add a statement of the form

```
if (is .eq. 6) call e(idata,data,x,y,S6)
```

just before "RETURN" in the subroutine "c" of the program DPDP. The second statement in this subroutine must be changed also by adding S6 to the set of external functions; i.e., this statement should be changed from

```
external  S1,S2,S3,S4,S5
```

to

```
external  S1,S2,S3,S4,S5,S6 .
```

IV. Programs Source Listing

```

c dppre.for - A program to prepare input data file for
c             the main program (dpep.for) (Detector
c             Performance Evaluation Program).
c
c
c Inputs - A set of numbers to be delivered through (tty)
c          as answers to questions. The format of each
c          number appears next to the question. This set
c          of numbers determines the detector to be tested,
c          the noise distribution function, the signal,
c          and the thresholds.
c
c
c Outputs - A file which contains all the above information
c           and serves as an input for dpep.for. File
c           format is (5I5,5f10.6). The name of this file
c           is (dpin.data).
c
0001      dimension idata(5),data(5)
0002      call setfil (1,'dpin.data')
c
c      This section determines the detector to be tested. The
c      answer to this question is an integer between 1 and 10
c      detector no.          detector's name
c
c      1 .      Linear detector.
c      2 .      Correlator detector.
c      3 .      Limiter-Correlator detector.
c      4 .      M-detector for contaminated
c              normal class.
c      5 .      M-detector for P-point class.
c      6 - 10   Optional.
c
0003      write (6,1)
0004 1      format ('detector no. =? (i5)')
0005      read (5,2) idata(1)
0006 2      format (i5)
c
c
c      This section determines the noise distribution
c      function. The answer to the next question should be
c      an integer between 1 and 10 according to the following
c      table.
c
c      Distribution no.      Distribution
c      .      1      .      Normal.
c      .      2      .      Double-Exponential.
c      .      3      .      A component of a Log-
c                          normal distribution.
c      .      4      .      Contaminated-Normal,
c                          the contaminating dist
c                          is also normal with
c                          optional variance. The
c                          contamination ratio is
c                          10%.
c      .      5      .      Contaminated-Normal as

```

```
      c      .      .      .      in 4 except that the
      c      .      .      .      contaminating distribution
      c      .      .      .      is a Double-Exponential.
      c
0007      write (6,8)
0008 8      format ('noise=? (I5)')
0009      read (5,2) idata(2)
      c
      c
      c      The answer to the next question should be a real
      c      positive number. If the noise is of type 1 or 2, then
      c      this number determines the standard deviation. If the
      c      noise is of type 3, then it determines the standard
      c      deviation of log(x). If the noise is either of type
      c      4 or 5, then this number determines the standard
      c      deviation of the contaminating distribution.
      c
0010      write (6,95)
0011 95     format ('scale=? (f10.6)')
0012      read (5,94) data(1)
      c
      c
      c      The answer here should be a real number which gives
      c      the signal amplitude.
      c
0013      write (6,10)
0014 10     format ('sig. amp=? (f10.6)')
0015      read (5,94) data(2)
      c
      c
      c      This is to define the signal wave form. If the
      c      answer is < 2, the signal will be considered as a
      c      constant. The same signal waveform will serve as
      c      a reference signal for correlator and M detectors.
      c
0016      write (6,3)
0017 3      format ('sig=? (I5)')
0018      read (5,2) idata(3)
0019      if (idata(3) .le. 1) idata(3)=1
      c
      c
      c      This is to determine the sample size. It should
      c      be an integer between 1 and 30.
      c
0021      write (6,90)
0022 90     format ('sample size=? (I5)')
0023      read (5,2) idata(4)
      c
      c      This is to determine the number of thousands
      c      of runs.
      c
0024      write (6,92)
0025 92     format ('no. of runs =1000*? (I5)')
0026      read (5,2) idata(5)
      c
      c
      c      This section determines the threshold (s) of the
```

c test and the type of test to be performed. If the  
c upper threshold (tu) is equal to the lower threshold (tl),  
c then it is a one threshold test, and the program (dpep)  
c determines the probability that the test statistic is in  
c either side of the threshold as is the case in the  
c minimum probability of error criterion. If tu is not  
c equal to tl, the distance between tl and tu will be  
c divided into 500 equal intervals, and the results of  
c (dpep) will be in the form of a histogram.

c

```
0027      write (6,93)
0028 93      format ('Upper Threshold =? (f10.6)')
0029      read (5,94) data(3)
0030 94      format (f10.6)
0031      write (6,97)
0032 97      format ('Lower Threshold =? (f10.6)')
0033      read (5,94) data(4)
0034      tref=0.0
0035      if (idata(1) .ne. 5) go to 99
```

c

c

c This number determines the break point of the  
c nonlinearity in detector no. 5. This is the "a" parameter  
c in the class of P-point distributions.

c

```
0037      write (6,98)
0038 98      format ('a=? (f10.6)')
0039      read (5,94) data(5)
0040 99      write (1,100) (idata(i) , i=1,5)
0041      write (1,200) (data(i), i=1,5)
0042 100     format (5i5)
0043 200     format (5f11.5)
0044      end file 1
0045      end
```

```

c DPEP.FOR: A program to evaluate the performance
c   of detectors in the presence of different
c   signals and against optional noise densities.
c
c   This program includes in addition to the main
c   routine the three subroutines d(.),c(.) and e(.),
c   and utilizes different functions from the
c   two libraries (dped) and (dpesn). The first
c   library contains all detectors to be tested. The
c   second library contains different signals
c   and noise generators.
c
c
c   Input: File (dpin.data) which is the output of the
c   data preparation program (dppre.for). File
c   format is (5i5,5f10.6).
c
c
c   Output: A binary file called (dpep.data). This file consists
c   of one record of length 2012 and contains the number
c   of observation groups utilized, the upper and lower
c   thresholds and the probability of false alarm or
c   detection depending on whether the signal is present
c   or not. This file can be transformed to a formatted
c   file using the program (dppost.for).
c
c
c Main routine:
c
c   d1,d2,...,d10 is a set of external functions
c   from the library (dped) and defines the detector
c   to be tested according to the following list:
c   d1 linear detector.
c   d2 correlator detector.
c   d3 limiter-correlator detector
c   d4 M-detector for contaminated normal class
c   d5 M-detector for p-point class
c   d6-d10 optional
c
c   Depending on the value of i=idata(1), the main
c   routine calls the subroutine d(idata,data,di).
c
c
0001 external d1,d2,d3,d4,d5,d6,d7,d8,d9,d10
0002 dimension idata(5),data(5)
0003 call setfil (2,'dpin.data')
0004 read (2,99) (idata(i),i=1,5)
0005 read (2,97) (data(i),i=1,5)
0006 97 format (5f11.5)
0007 99 format (5i5)
0008 i=idata(1)
0009 if (i .eq. 1) call d(idata,data,d1)
0011 if (i .eq. 2) call d(idata,data,d2)
0013 if (i .eq. 3) call d(idata,data,d3)
0015 if (i .eq. 4) call d(idata,data,d4)
0017 if (i .eq. 5) call d(idata,data,d5)
0019 if (i .eq. 6) call d(idata,data,d6)

```

```
0021      if (i. eq. 7) call d(idata,data,d7)
0023      if (i. eq. 8) call d(idata,data,d8)
0025      if (i. eq. 9) call d(idata,data,d9)
0027      if (i. eq. 10) call d(idata,data,d10)
0029      end
```

```
      C
0001      subroutine d(idata,data,x)
      C
      C
      C Subroutine      d(idata,data,x):
      C
      C      idata and data are two arrays from the main
      C      routine.
      C      x is a dummy argument to be replaced by the
      C      call name of the detector under test.
      C      zn1,zn2,...,zn10 is a set of external functions
      C      from the library (dpesn) and defines the noise
      C      distribution to be utilized according to the
      C      following list.
      C      zn1 normal distribution
      C      zn2 double exponential
      C      zn3 a component of a log-normal distribution
      C      zn4 contaminated normal, the contaminating
      C      distribution is also normal with
      C      optional variance. The contamination
      C      ratio is 10%.
      C      zn5 contaminated-normal as zn4 except the
      C      contaminating distribution is a
      C      double-exponential.
      C      zn6-zn10 optional
      C
      C      Depending on the value of j=idata(2), this
      C      subroutine calls the subroutine c(idata,data,x,zn(j)).
      C
      C
0002      external zn1,zn2,zn3,zn4,zn5,zn6,zn7,zn8,zn9,zn10
0003      dimension idata(1),data(1)
0004      in=idata(2)
0005      if (in .eq. 1) call c(idata,data,x,zn1)
0007      if (in .eq. 2) call c(idata,data,x,zn2)
0009      if (in .eq. 3) call c(idata,data,x,zn3)
0011      if (in .eq. 4) call c(idata,data,x,zn4)
0013      if (in .eq. 5) call c(idata,data,x,zn5)
0015      if (in .eq. 6) call c(idata,data,x,zn6)
0017      if (in .eq. 7) call c(idata,data,x,zn7)
0019      if (in .eq. 8) call c(idata,data,x,zn8)
0021      if (in .eq. 9) call c(idata,data,x,zn9)
0023      if (in .eq. 10) call c(idata,data,x,zn10)
0025      return
0026      end
```

```
      C
      C
0001      subroutine c(idata,data,x,y)
      C
      C      Subroutine c (idata,data,x,y)
      C
      C      idata, and data are two arrays from routine
      C      main through routine d(.....).
      C      x is a dummy argument to be replaced by the
      C      call name for the detector under test.
      C      y is a dummy argument to be replaced by the
      C      call name for the noise distribution function
      C      from subroutine d.
      C      s1,s2,...,s5 is a set of external functions
      C      from library (dpsn) and defines the signal wave
      C      form. It also defines the reference signal
      C      for some detectors. s1 is a constant signal and
      C      others are optional.
      C
      C      Depending on the value of k=idata(3), this
      C      subroutine calls the subroutine e(idata,data,x,y,sk).
      C
      C
0002      external s1,s2,s3,s4,s5
0003      dimension idata(1),data(1)
0004      is=idata(3)
0005      if (is .eq. 1) call e(idata,data,x,y,s1)
0007      if (is .eq. 2) call e(idata,data,x,y,s2)
0009      if (is .eq. 5) call e(idata,data,x,y,s5)
0011      if (is .eq. 4) call e(idata,data,x,y,s4)
0013      if (is .eq. 3) call e(idata,data,x,y,s3)
0015      return
0016      end
```

```

      C
      C
0001      subroutine e(idata,data,x,y,z)
      C
      C Subroutine e (idata,data,x,y,z):
      C
      C This is the main routine in the test
      C package where actual detector test is performed.
      C
      C idata,data,x and y are as described in the subroutine
      C c.
      C z is a dummy argument to be replaced by the call
      C none for the signal wave form from subroutine c.
      C
      C r=data(1), defines the standard deviation of the
      C noise distribution if it is zn1 or zn2 and defines
      C the standard deviation of log(x) if the distribution
      C is zn3. It defines the standard deviation of
      C the contaminating distribution in case of zn4
      C and zn5.
      C
      C t=data(2) is the signal amplitude.
      C
      C tu=data(3) and tl=data(4) defines the
      C upper and lower thresholds respectively. If tu=tl
      C the test to be performed is a one threshold test.
      C If tu is not equal to tl the output of
      C the test is a histogram with the distance
      C between tl and tu divided into 500 equal
      C intervals.
      C
      C tref=data(5), is utilized only with d5 to
      C define the break points of the nonlinearity.
      C
      C n=idata(4) is the sample size.
      C
      C n1=idata(5) is the number of thousands of times
      C the test to be performed.
      C
0002      dimension idata(1),data(1),ox(30),tr(500),ref(30)
0003      call setfil (1,'dpep.data')
0004      define file 1 (1,2012,u,kx)
0005      i=idata(1)
0006      r=data(1)
0007      t=data(2)
0008      tu=data(3)
0009      tl=data(4)
0010      tref=data(5)
0011      n=idata(4)
0012      n1=idata(5)
0013      99      su=tu-tl
0014      do 100 n=1,n
0015      100      ref(n)=z(n)
0016      do 101 n2=1,n1
0017      do 102 n=1,1000
0018      do 103 id=1,n
0019      103      ox(id)=y(r)

```

```
0020      if (t .eq. 0.0) go to 105
0022      do 104 id=1,n
0023 104    ox(id)=ox(id)+t*ref(id)
0024 105    st=x(ox,n,ref,tref)
0025      if (su .eq. 0.0) go to 106
0027      if (st .ge. tu) go to 200
0029      if (st .le. t1) go to 102
0031      k=int(((st-t1)/(tu-t1))*500+1)
0032      go to 210
0033 200    k=500
0034 210    tr(k)=tr(k)+1
0035      go to 102
0036 106    if (st .gt. tu) go to 107
0038      tr(1)=tr(1)+1
0039      go to 102
0040 107    tr(2)=tr(2)+1
0041 102    continue
0042 101    write (1'1) n2,t1,tu,(tr(j), j=1,500)
0043      end file 1
0044      return
0045      end
```

```
c DPPOST.FOR
c
c   A program to transform the binary file (dpep.data)
c   to a formatted file.
c
c
c Input:
c   (dpep.data) a one record binary file of length
c   (2012). This is the output of the program (dpep.for).
c
c
c Output:
c   ((dpout.data) a formatted file. First line is the
c   number of runs. The rest of the file consists of two columns
c   containing threshold (T) vs. probability of exceeding this
c   threshold.
c
c
0001      dimension t(500),b(500)
0002      call setfil (1,'dpep.data')
0003      define file 1 (1,2012,u,kx)
0004      call setfil (2,'dpout.data')
0005      read (1'1) n,t1,tu,(t(i),i=1,500)
0006      dn=n*1000.0
0007      write (2,1000) dn
0008 1000  format('number of runs = ',f11.1)
0009      write (2,1500)
0010 1500  format('   Threshold (T)   ',15x,'p(x>T)')
0011      if (tu .eq. t1) go to 100
c
c
0013      do 10 i=1,499
0014      j=500-i
0015 10    t(j)=t(j)+t(j+1)
0016      dt=(tu-t1)/500.0
0017      do 20 i=1,500
0018      t(i)=t(i)/dn
0019 20    b(i)=t1+(i-1)*dt
0020      do 30 i=1,500
0021 30    write (2,2000) b(i),t(i)
0022 2000  format (f10.5,10x,f20.8)
0023      go to 200
c
c
0024 100   t(2)=t(2)/dn
0025      write(2,2000) t1,t(2)
0026 200   continue
0027      end file 1
0028      end file 2
0029      end
```

```
c
c DPED.FOR
c   A set of detector functions to support the
c   program DPEP.
c
c
c   For any of these functions xd is the observation array,
c   n is the sample size and ref is the reference signal
c   array. The variable tref is used only with the detector
c   function d5 as the break point of the nonlinearity.
c
c
c
0001   real function d1(xd,n,ref,tref)
c
c   Linear Detector
c
0002       dimension xd(1)
0003       yd=0.0
0004       do 201 ii=1,n
0005 201   yd=yd+xd(ii)
0006       yd=yd/n
0007       d1=yd
0008       return
0009       end
```

```
0001      real function d2(xd,n,ref,tref)
      c
      c Correlator Detector
      c
0002      dimension xd(1),ref(1)
0003      yd=0.0
0004      do 211 i1=1,n
0005 211  yd=yd+xd(i1)*ref(i1)
0006      yd=yd/n
0007      d2=yd
0008      return
0009      end
```

```
0001      real function d3(xd,n,ref,tref)
      c
      c Limiter-Correlator Detector
      c
0002      dimension xd(1),ref(1)
0003      yd=0.0
0004      do 221 ii=1,n
0005      if (abs(xd(ii)) .ge. 1.14) go to 222
0007      yd1=xd(ii)*ref(ii)
0008      go to 223
0009 222  yd1=1.14*ref(ii)*sign(1.,xd(ii))
0010 223  yd=yd+yd1
0011 221  continue
0012      d3=yd/n
0013      return
0014      end
```

```
0001      real function d4(xd,n,ref,tref)
      c
      c M-Detector for Contaminated Normal Class
      c
0002      dimension xd(1),ref(1)
0003      eu=8.0
0004      el=-8.0
0005      en=0.0
0006      do 231 ii=1,20
0007      dxt=0.0
0008      do 232 jj=1,n
0009      ens=en*ref(jj)
0010      dx=xd(jj)-ens
0011      if (abs(dx) .ge. 1.14) dx=1.14*sign(1.,dx)
0013 232      dxt=dxt+dx*ref(jj)
0014      if (dxt) 234,235,236
0015 234      eu=en
0016      go to 237
0017 236      el=en
0018 237      en=(eu+el)/2.0
0019 231      continue
0020 235      d4=en
0021      return
0022      end
```

```
0001      real function d5(xd,n,ref,tref)
      c
      c
      c M-Detector for P-Point Class
      c
0002      dimension xd(1),ref(1)
0003      eu=8.0
0004      el=-8.0
0005      en=0.0
0006      do 241 ii=1,20
0007      dxt=0.0
0008      do 242 jj=1,n
0009      ens=en*ref(jj)
0010      dx=xd(jj)-ens
0011      if (abs(dx) .le. tref) go to 243
0012      dx=0.663*sign(1.,dx)
0013      go to 244
0014      243 dx=(dx/(1.718*tref))
0015      dx=sin(dx)/cos(dx)
0016      dxt=dxt+dx*ref(jj)
0017      244 continue
0018      242 if (dxt) 245,246,247
0019      245 eu=en
0020      go to 248
0021      247 el=en
0022      248 en=(el+eu)/2.0
0023      241 continue
0024      246 d5=en
0025      return
0026      end
0027
```

UNIX fortran iv v01-11 source listing

page 001

```
0001      real function d6(xd,n,ref,tref)
0002      d6=0.0
0003      end
```

UNIX fortran iv v01-11 source listing

page 001

```
0001      real function d7(xd,n,ref,tref)
0002      d7=0.0
0003      end
```

UNIX fortran iv v01-11 source listing

page 001

```
0001      real function d8(xd,n,ref,tref)
0002      d8=0.0
0003      end
```

UNIX fortran iv v01-11 source listing

page 001

```
0001      real function d9(xd,n,ref,tref)
0002      d9=0.0
0003      end
```

UNIX fortran iv v01-11 source listing

page 001

```
0001      real function d10(xd,n,ref,tref)
0002      d10=0.0
0003      end
```

```

C
C DPESH.FOR
C   A library contains signal waveforms and noise
C   generator functions required by the program
C   DPEP.
C
C   for all signal wave form functions the parameter
C   js represents the sample order.
C
C   For all noise functions the parameter an (or a)
C   represents the scale parameter from the program
C   DPPRE.
C
C
0001      real function sl(js)
C
C   Constant Signal
0002      xs=1.0
0003      sl=xs
0004      return
0005      end
```

UNIX fortran iv v01-11 source listing

page 001

```
0001      real function s2(a)
0002      xa=100.0*a
0003      s2=xa
0004      return
0005      end
```

UNIX fortran iv v01-11 source listing

page 001

```
0001      real function s3(js)
0002      s3=0.0
0003      end
```

UNIX fortran iv v01-11 source listing

page 001

```
0001      real function s4(a)
0002      s4=0.0
0003      end
```

UNIX fortran iv v01-11 source listing

page 001

```
0001      real function s5(a)
0002      s5=0.0
0003      end
```

```
0001      real function zn1(an)
      c
      c Normal Distribution
      c
0002      integer flag
0003      if (flag .eq. 12345) go to 420
0005 410    xzn=2.0*an(0,0)-1.0
0006      yzn=2.0*ran(0,0)-1.0
0007      zzn=xzn*xzn+yzn*yzn
0008      if (zzn .ge. 1.0) go to 410
0010      zzn=sqrt(-2.0*alog(zzn)/zzn)
0011      zn1=xzn*zzn*an
0012      yzn=yzn*zzn*an
0013      flag=12345
0014      return
0015 420    zn1=yzn
0016      flag=0
0017      return
0018      end
```

```
0001      real function zn2(an)
      c
      c Double Exponential
      c
0002      zn2=sign(1.,ran(0,0)-0.5)*an*-0.70710678*log(ran(0,0))
0003      return
0004      end
```

```
0001      real function zn3(an)
      c
      c
      c A Component from a Lognormal Distribution
      c
0002      zn3=exp(zn1(an))*cos(6.2831853*ran(0,0))
0003      return
0004      end
```

```
0001      real function zn4(an)
      c
      c Normal Contaminated by Normal.
      c
0002      zz=ran(0.0)
0003      if (zz .ge. 0.1) go to 441
0004      zn4=zn1(an)
0005      return
0006      441  zn4=zn1(1.0)
0007      return
0008      end
0009
```

```
0001      real function zn5(an)
      c
      c Normal Contaminated by Double Exponential
      c
0002      zz=ran(0,0)
0003      if (zz .ge. 0.1) go to. 451
0005      zn5=zn2(an)
0006      return
0007 451  zn5=zn1(1.0)
0008      return
0009      end
```

UNIX fortran iv v01-11 source listing

page 001

```
0001      real function zn6(a)
0002      zn6=0.0
0003      end
```

UNIX fortran iv v01-11 source listing

page 001

```
0001      real function zn7(a)
0002      zn7=0.0
0003      end
```

UNIX fortran iv v01-11 source listing

page 001

```
0001      real function zn8(a)
0002      zn8=0.0
0003      end
```

UNIX fortran iv v01-11 source listing

page 001

```
0001      real function zn9(a)
0002      zn9=0.0
0003      end
```

UNIX fortran iv v01-11 source listing

page 001

```
0001      real function zn10(a)
0002      zn10=0.0
0003      end
```

## References

1. J. W. Tukey, "A Survey of Sampling from Contaminated Distributions", in Contributions to Probability and Statistics (Harold Hotelling) Vol. I, Olkin et al., Eds. Stanford, Calif. Stanford University Press, 1960, pp. 448-485.
2. F. R. Hampel, "Contributions to the Theory of Robust Estimation", Ph.D. Dissertation, Univ. of Calif., Berkeley, 1968, pp. 1-6.
3. P. A. Bello and R. Esposito, "A New Method for Calculating Probabilities of Errors Due to Impulsive Noise", IEEE Trans. Comm. Tech., vol. Com-17, No. 3, June 1964, pp. 368-379.
4. A. D. Vatt and E. L. Maxwell, "Measured Statistical Characteristics of VLF Atmospheric Noise", Proceeding of IRE, Jan. 1957, pp. 55-62.
5. D. R. Rlose and L. Kurz, "A New Representation Theory and Detection Procedures for a Class of Non-Gaussian Channels", IEEE Trans. Comm. Tech., Vol. Com-17, No. 2, April 1969, pp. 225-234.
6. P. J. Huber, "A Robust Version of the Probability Ratio Test", Ann. of Math. Statist., Vol. 36, Dec. 1965, pp. 1753-1758.
7. S. A. Kassam and J. B. Thomas, "Assymptotically Robust Detection of a Known Signal in Contaminated Non-Gaussian Noise", IEEE Trans. Inform. Theory, Vol. IT-22, Jan. 1976, pp. 22-26.

8. G. V. Trunk, "Trimmed Mean Detector for Noncoherent Distributions", NRL Report 6997, Dec. 1969.
9. A. H. El-Sawy and V. D. VandeLinde, "Robust Detection of Known Signals in Noise, IEEE Trans. Inform. Theory, Vol. IT-23, No. 6, pp. 722-727, Nov. 1977.
10. R. D. Martin and S. C. Schwartz, "Robust Detection of a Known Signal in Nearly Gaussian Noise," IEEE Trans. Inform. Theory, Vol. IT-17, pp. 50-56, Jan. 1971.